

"The web has a crying lack of machine-processable information"

Tim Bray, co-inventor of XML

NEW GENERIC LANGUAGE ADDRESSES SEMANTIC INTEROPERABILITY, DATABASE INTEGRATION, CHALLENGES XML, PRESENTS NEW PERSPECTIVES ON ONTOLOGIES AND SEMANTIC WEB

INTRODUCTION

Could the Holy Grail of Information Technology be a global new language that can represent everything in a uniform way, beyond natural languages and measurement systems? Its base lexicon, already comprising 450,000 ' ' in their basic and variant forms, will be open-sourced, together with tools to create new basic words and word aggregates, thesauri, and contexts.

Deliberately ignoring protocols and open standards, this David challenges the XML Goliath. Its goal has been to address data fragmentation and apply reasoning on its constructs, while avoiding the shortcomings and redundancies of current technologies such as RDF and the United Nations UNL effort.

The research has resulted in a unified way of representing information such as data in databases, business objects, processes, web and document contents, metadata, and ontologies. The extensible semantic markup associated with this language can be understood worldwide and is three times shorter than XML. It may even help in indexing binary data and searching it by using keywords, including pictorial and sound information.

Life Functions, Business Processes

All humans share the same physical and psychological needs. To satisfy their need to communicate they use the same pool of concepts and translate them into words. They use documents that contain the same type of information, e.g. shipping documents in Tokyo or Washington include the same fields, such as sender, recipient, and content. However, the words used are different, both visually and phonetically. Moreover, a word in one language may be represented by a longer phrase in another.

Elements of natural languages are used to create computer languages, contents in databases, applications, and web pages. Internally, these are zeroes and ones. Computer programs operate on several data types such as integer, float, string of characters, or collections or aggregations of them, creating 'objects'.

Business Objects Are Named After Words Found in Occupational Vocabularies

Business objects are reusable objects or server components that encapsulate business logic. They are tangible entities within an application that developers create, access, and manipulate; the shipping document mentioned

above is a business object. They contain all information necessary for computations and presentation: an invoice will include the invoice number, purchase date, client name and address, client ID, articles sold, etc.

Simple business objects (" ", " ", " ") serve as building blocks for **complex** business objects (e.g. " "). To qualify these objects, we use **words**, or aggregates of words found in the common occupational vocabulary of a group of people (Payroll, Accounting). The words used in Business Processes also originate directly or indirectly from a natural language¹.

Are we Using a Foreign Language?

In most corporations, software architects use different names for the same business entities in different applications, e.g. **“Date_Created”** and **“DateCreated”**. Thus databases are cluttered with similar information that no tool **easily** succeeds in identifying as being similar! If computers do not ‘understand’ that the above two names represent the same business object, isn’t giving different names to the same business objects or processes equivalent to using a foreign language? Shouldn’t we unify the names of business objects so that computers know that the above two names are synonyms, as are “ ” and “日期创造”?

The problem becomes more serious when companies have to combine operations and IT applications, e.g. in the case of a merger. In fact, many companies dedicate over fifty percent of their IT operations budget to interfacing with legacy systems or external finance, supply chain, CRM and other mission critical systems!

Data Fragmentation ≡ Lack of Integration

Multiple vendors offer numerous representation schemes with different names for the same business objects, or textual data in foreign languages; this entails data fragmentation², which decreases the size of datasets used for data mining, and consequently reduces the accuracy of predictive analytics³. Thus, competitive advantages are lost, such as subject matter expertise, which is perhaps the most important asset of an organization⁴.

Conceptual Interoperability and Metadata

The sole culprit to all these **deficiencies** is the lack of global, conceptual, semantic interoperability.

Last month, the Object Management Group (OMG, Model Driven Architectures) heralded that they would produce a proof of concept of their “Model Driven Message Interoperability” (MDMI) to implement global semantic interoperability. However, using metadata does not provide conceptual or semantic interoperability, but merely reaches technical interoperability (level 6 versus level 1, as per **“Levels of Conceptual Interoperability Model (LCIM), developed at the Virginia Modeling Analysis & Simulation Center**).

An overview of frameworks claiming to bring interoperability, e.g. Open Group Architecture’s TOGAF, OASIS’s Open Document Format, MIT’s M Language, Cycl, Framenet, and Metanet, shows that some of them define business entities but ignore any description that goes beyond English, **or fail in establishing correspondences between identical business objects expressed in various natural languages**.



Figure I – Conceptual Semantic Interoperability, the Common Vector

Does XML Address Global Interoperability?

XML appeared at a time when proprietary formats had invaded the IT field, and it provided a uniform way to present information and design interfaces between applications; however, it has not helped in consolidating semantically identical data across languages, or even in one language, as shown by the examples below:

- The proposition: `<EMPLOYEENO>111-222-333</EMPLOYEENO>`
is different from: `<EmployeeNo>111-222-333</EmployeeNo>`

Clearly, XML does not recognize ‘identical’ names. Again, using different names for the same object in the ‘same’ natural language is detrimental to semantic interoperability and integration.

- Also, XML fails to understand that the proposition:
`<temperature unit="Fahrenheit" value="98.6" />`
is identical to: `<temperatura unità="Fahrenheit" valore="98.6" />`
and to: `<temperatura unità="Celsius" valore="37" />`

This last example also suggests that quantities should be expressed independently from measurement systems (imperial, metric), resulting in a generic representation of data, thus achieving a higher degree of integration.

- When invoking web services, developers compose a SOAP message with the parameters required by WSDL, but yet, this technology does not provide interoperability at a name level; and agreeing on SOAP provides little actual interoperability between applications.

- A look at the Dublin Core metadata shows that its set of tags [Title, Creator, Subject, etc.] is expressed solely in English, as all standards (OAI/DC, IEEE/LOM, etc.) are English-centric.

English has become the ‘universal’ language, yet computers using English (or Italian, or Arabic) words cannot process them because they have of knowing what they represent. And various naming conventions hide the fact that using different names to represent similar business objects in some applications makes them inadequate for processing and/or integration.

XML does not implement global semantic interoperability. Nor does it help prevent data fragmentation. Knowing these facts, isn’t it worth crafting a technology that would give a unique name to all similar items across languages?

XML is touted to offer human readability, but to which humans? Only the subset of humans who can understand the language used in the document? Can everyone understand `<採買>香水</採買>`? Fortunately, new technologies are challenging XML. JSON, for instance, has more proponents every day around the world.

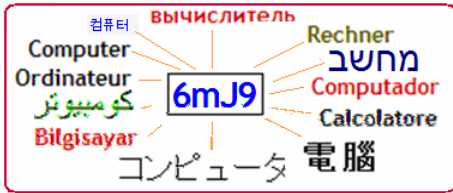
- **John Sowa**, IBM fellow and inventor of conceptual graphs, states:

- **Oliver Goldman, Jon Ferriolo, and Larry Masinter** from Adobe Systems, Inc., members of the W3C on the Binary Interchange of XML Infosets, stated the following:

- Quoting **Terence Parr**, Chief scientist at jGuru.com and author of TML (Terence’s Markup Language):

A Generic Representation

All languages are based on concepts⁵ which are common to all humans and allow one to translate into other languages by adapting words and syntactic constructs in a source language to the equivalent in a target language without changing semantics⁶. This suggests that the only way to attain conceptual interoperability is to map equivalent concepts in various languages with the same word, as shown below*:



*Expressing a concept with one word in **one** language and two or more words in another, e.g. “monter”, “go up” or “上去”, or the absence of an equivalent in a foreign language, **are** beyond the scope of this presentation.

Figure II – A word and its counterparts in various [local] natural languages

KODAXIL

Kodaxil (Knowledge, Object, Data, Action, eXtensible Interoperable Language) is a

. It does not use XML-based technologies, but substitutes words in natural languages with ‘smart universal words’ aware of their grammatical role and their length (parts of speech, semantic invariants across languages⁷). Each word is a base64-encoded proxy to the same word in various natural languages. These words are assembled to form an Interlingua for generic representation⁸.

The base lexicon currently comprises almost 450,000 words in their neutral and variant forms¹⁰. It also includes reserved words that help compose markup documents, structures, or explicit tags. When all variant forms –especially verbs in their many tenses and modes– are encoded, the base lexicon will comprise 8 million words. This makes it one of the existing corpora and multilingual dictionaries.

As one bit is used to inform parsers about word length; this allows the base lexicon to contain $2^{(24 - 1)} = 8388608$ words; in addition, one can create lexicons, thesauri and contexts at will.

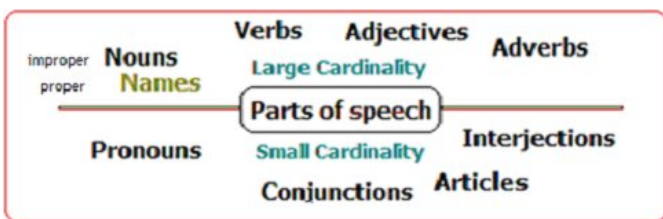


Figure III – Parts of speech, sense and denotation are semantic invariants

Kodaxil lexicon	English Thesaurus
ul4	uL40 "To Be"
	uL4e "am"
	uL4f "is"
	uL4g "are"
	uL4J "was"
	uL4K "were"
	etc.

Figure IV – Derivative forms & encoding, never losing track of conceptual information

Systemic Decisions in the Design of a “Universal Word”:

The Universal Word (UNL) * effort was undertaken in 1996 by the United Nations. A short overview of its ‘nominal concepts’ shows that it categorizes things, tangibles, and derives objects based on this primal ontology.

- For one, it may be hard to define the type of an entity; e.g. one that is at the boundary of living and inert matter, such as a virus, which is not considered a “living” entity, inert as a ‘virion’, but can reproduce within a host.

It is also difficult to distinguish between countable or uncountable, abstract and tangible concepts when they are represented by the same word, such as the word ‘youth’ (as in ‘in my youth I traveled...’ versus ‘twenty youths were part of the effort...’).

Deriving from the mutually exclusive properties of UNL, due to the primal ontology they opted for, would result in an empty set! The conclusion is that in order to represent all possible semantic hues, it is wiser to treat entities on a per-usage basis, and offer the possibility to link an object to more than one category, eventually disambiguating using Bayesian or other methods (using HMM, returned value of function, GA, etc.). UNL was discarded because it did not offer this capability, and a decision was made to use the **{word-thesaurus-context}** to manage words, and objects thereafter:

- Words store a minimum amount of information: lexical class and length, for parser use.
- Grammatical information, which varies between natural languages, e.g. gender (masculine, feminine, neuter), is added to a thesaurus.
- A context stores all possible usage cases, sense / denotation, relations and lineage, rather than implying that a “thing” derives from some initial object. This is akin to normalization in database modeling.
- For object definition within an application, a context gathers all possible relationships
- For event definition it gathers conditions of occurrence, and so on.

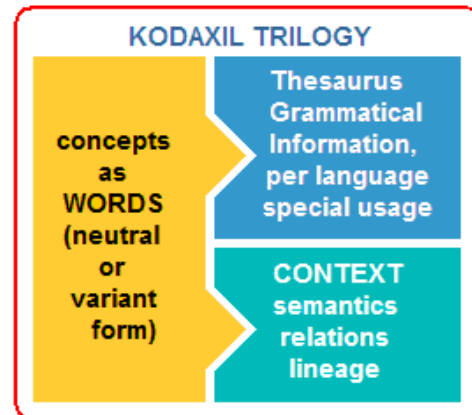


Figure V – Organization of conceptual information

Key words, Namespaces, Persistence

Each word acts as a unique key to an entry in a thesaurus or a context (also known as auxiliary file, or knowledge base). Thesauri bring natural-language-specific grammatical information, including gender, attributes, and constraints, to disambiguate the term when necessary, as well as usage cases. A base context gathers information about parts of speech, but one can create as many contexts as desired.

Verbs may be transitive, intransitive, or di-transitive. Nouns and adverbs may belong to more than one category. As this semantic information is valid worldwide, these parts of speech are stored into contexts, as opposed to thesauri which contain natural-language-specific information.

Each object or construct is associated either with a whole context or with an entry in a larger context. In markup documents and in applications, contexts can be used as *namespaces* in order to disambiguate either some specific usage of a word, or determine where the definition of the word or object is located. Contexts may eventually contain a singleton such as URI, so they can represent legacy systems.

Contexts provide semantics - sense and denotation, synonyms, and a list of relationships and direct parents: for instance, ‘man’ ‘is A’ mammal, ‘has A’ gender: male (and mammal ‘is A’ vertebrate, etc.), all expressed using Kodaxil words. A context is primarily a hash table where keys are reserved words of the language, and values are anything from hash tables to lists or discrete values. As one can ‘persist’ (serialize) hash tables as arrays of bytes, these can be stored in a database. A flat file version exists, but it is much slower to access. The beauty of contexts is that they let express all that topic maps specify and more, but also in many languages at once, as Kodaxil words are universal.

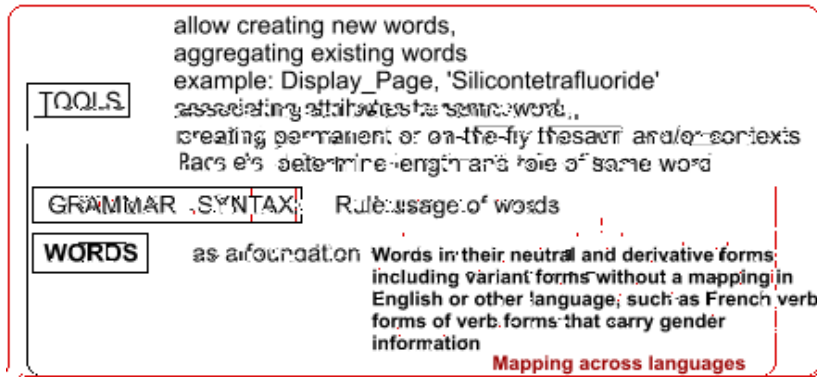


Figure VI. – Smart Universal Words as the foundation of Kodaxil

Inference, Object-orientation, Multiple Inheritance

Allowing an object to derive from one or more objects for lineage information avoids the ontology error evoked above. Allowing contexts to contain named relation(s) offers a built-in object-orientation mechanism – including multiple inheritances, allowing one to reason on the constructs using words or objects.

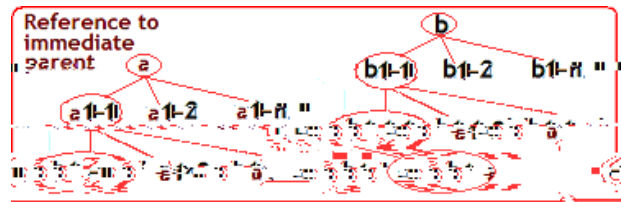


Figure VII. – Link to direct parent(s)

The word or object described here is both an a1-1-1 (is A) and a b1-1-1-2. Its context only needs to store a reference to its immediate direct parents a1-1 and b1-1-1, because the context for a1-1 contains ‘a’ as direct parent, and the context for b1-1-1 contains ‘b1-1’ as direct parent.

The augmented knowledge base established by contexts makes inference possible by using a backward chaining mechanism, and allows finding all children / leaves in a forward chaining operation –performing a breadth and depth search– and process minimax and other algorithms.

Is A: ‘Man’ or ‘woman’ ‘human’, as their context contains ‘human’ [‘Man’ ‘is A’ human] and information about gender. ‘Human’ ‘is A’ ‘mammal’, information stored in the base context is for ‘human’. In the context entry for ‘mammal’, ‘vertebrate’ is found, which, two or three more steps up, chains to ‘life’, and to ‘mortal’, an adjective linked to the antonym, found in the context for ‘life’.

The potential set of inferences about ‘Socrates’ - a man -, provides, among others, the information that ‘Socrates is mortal’, which is based on first-order logic. As Kodaxil constructs can be embedded and stacked, an expression can be used as a parameter to be evaluated at unification time, and this feature allows Kodaxil to implement second- to n-order logic.

Has A: ‘Mammal’ in turn ‘has A’ ‘mammalBody’ (property), so ‘man’ ‘has A’ body as it ‘is A’ (inherits from) mammal. Contexts allow building very large knowledge bases and ontologies (definitions), in this generic language, for use around the world. Again, Kodaxil internal representation is “bodyMammal” since in this case “mammal” is an attribute of body which allows defining a ‘Class’.

The context for “body” contains other usage-based information, as we may use “body” as the body of a message, body of a voice, etc., all expressed using sentences of these universal words, as well as more direct parents.

Word Length

Each word of the base lexicon comprises four sextets, consisting of pure printable ASCII characters. Kodaxil parsers read each bit. If this bit’s value is ‘0’, then it is a word of the base lexicon; if its value is ‘1’,

then the parser ‘knows’ where to find the continuation that informs it about offset and object type (word greater than 4 sextets or construct).

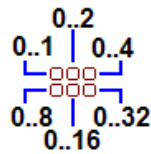


Figure VIII – Encoding of a sextet

Grammar and Syntax

Grammar and syntax provide rules for aggregation of existing words or creation of new ones, offering the possibility to add new entries to thesauri and auxiliary files (knowledge bases) making this language extensible ad infinitum. In-depth coverage of its grammar, known as **Kodaxiom** (Kodaxil Object Definition And Ontology Modeler) is beyond the scope of this introduction.

Sentences

As in natural languages, sentences in Kodaxil consist of sequences of these ‘universal words’ [**regular expression**], or an **ANTLR EBNF tree**, they may mix words of the (already base64-encoded) language and base64-encoded ‘raw’ text in some natural languages, and/or base64-encoded multimedia objects preceded by their mime type, allowing one to create multimedia documents.

After this introduction to the structural components of Kodaxil, let us look at objects used in IT, their definitions and selected Kodaxil usage cases.

Semantic Objects in IT

The following examples represent most of the semantic objects found in IT:

The President goes to China

TOMORROW, WIND WILL BE EAST 15 TO 20 MPH WITH GUSTS TO 25 MPH. TEMPERATURE WILL REMAIN IN THE LOW 50S FOR NEW HAMPSHIRE

A script, or an algorithm, uses the imperative mode. For instance, ‘**b2 – 4ac**’ can be represented by a sequence of statements:

1/ MULTIPLY ‘b’ BY ‘b’

2/ MULTIPLY ‘a’ BY ‘c’

3/ SUBTRACT THE RESULT OF 2/ FROM THE RESULT OF 1/...

As for requirements, so critical to the software development process:

The ‘Save’ button must be enabled for Managers, it must be disabled for other staff

The view must be sorted by employee name, in ascending order

These semantic objects share a common feature; information, knowledge, data, facts, rules, requirements, specifications, and configurations can be expressed using **words**, and understood owing to grammar and syntax. Even numeric data can be embedded within phrasal context, providing information such as the type of data processed, as shown below:

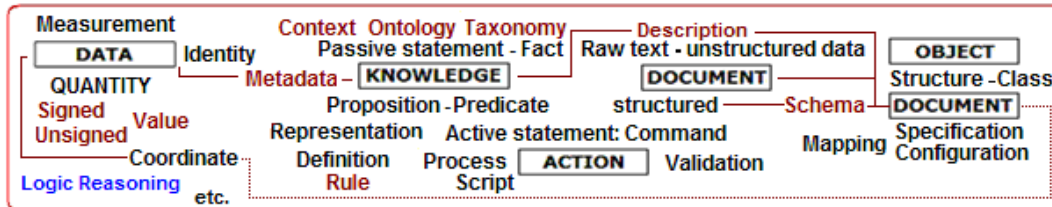


Figure IX – IT Objects, and realms of information

Organization in Trees and Case Study: Quick - ‘Raw’ / ‘Unmanaged’ – Translation

Simple rules allow for providing a of Kodaxil ‘raw’ strings, as Business Objects, which use a neutral grammatical form, do not need translation rules. The usual ‘managed’ representation strips the text from all local parts while conserving all semantic elements that allow translation to other languages. It does this by representing sentences, questions, and predicates using Kodaxil words, in the sequence determined by Kodaxil grammar and syntax.

Example:

1. Article; form: definite; person: singular, masculine [Words must include gender information, even though it may not be used in some languages, to provide a counterpart (key) in languages that use it.]
2. Verb = ‘to go’; tense = ‘present’; person: third, singular, masculine
3. Connective implying direction
4. Country Name
5. The string contains all elements that allow translation to any language, as shown in Figure X.

- In the first pass (where the ‘Equalizer’ finds the boundary of sentences), text in a natural language is parsed and converted to Kodaxil words.
- Then, a target language is chosen and the corresponding thesaurus is used (the ‘Renderer’).

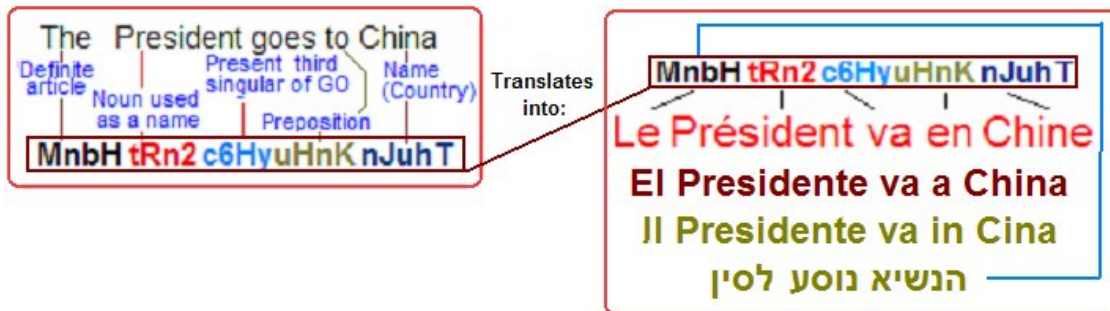


Figure X Quick (“Raw”) Representation and Translation of the Same Text

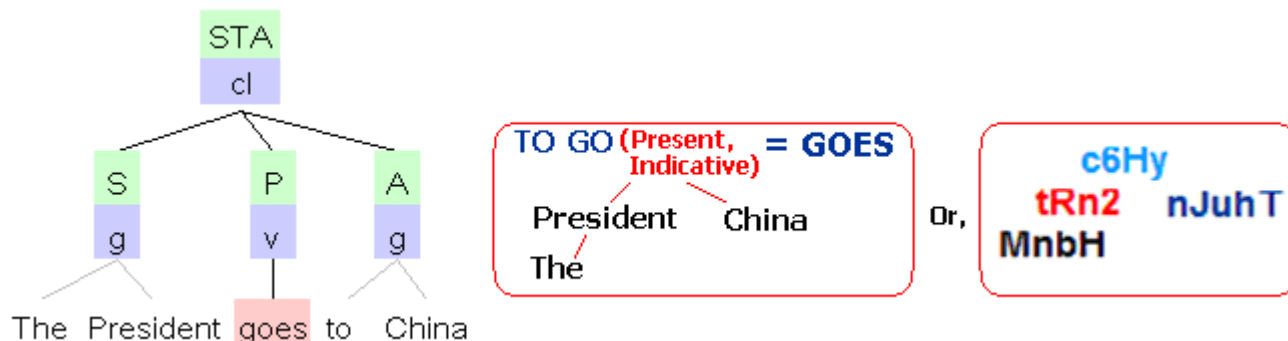


Figure XI – ‘Managed’ representation - Organization of Subject and Predicate

Disambiguation

Case 1:

This question might seem illogical, but it makes sense if uttered by a priest.

Case 2:

Since you are here, help me carry the piano that I have owned since 1972.

Each instance of “since” has a different role: the first is an adverb, while the second is a conjunction.

‘Since’ may also be a preposition. Note that causation / causality involves a sequence/an orchestration in time (if/because...then), and the various roles of ‘since’ do reflect this fact.

Note that most word sense disambiguation methods can be applied to Kodaxil words.

MARKUP: IMPLICIT, EXPLICIT

Markup has existed since the inception of writing, under various forms, and not only the form exhibited by SGML and XML. For instance, in a printed text, the might consist of carriage returns and indentations, **as shown in the example below:**

Entrées

Saumon fumé dans une coupe de melon au Porto

Escargots de Bourgogne sur lit de morilles

Plat Principal

Sanglier aux bolets d’Auvergne

Cotes de boeuf à la moëlle sauce marchand de vin

Plateau de Fromages

Desserts

Crème brûlée

Iles flottantes

;

In digital content, markup might also consist of specific words to inform parsers about the layout of the document. On the other hand, may use information such as ‘length-aware’ words, for parsers to ‘understand’ the syntax.

Illustrative Cases

Kodaxil and XML Representations of the Same Information

This XML document uses 231 characters and can be understood by English speakers only. The Kodaxil counterpart is a 71-character, pure printable ASCII string:

Fce83mlJjkeNhjV7t6yag3p0jkehM6USm9obg==n88rkj4rhM6UQXJtc2J5n88ri8uxi8ux

- **It is 72% smaller than its XML counterpart**
- It can be understood worldwide using client tools to decode it
- It is platform and system (metric, imperial) independent.

However, composing it is easy using Kodaxil tools, as Kodaxil be read by humans too

CustomerName

INDENT:NAME

INDENT:FirstName:John, LastName:Armsby

UNINDENT

UNINDENT

Kodaxil Reserved Words

After inserting spaces between words in the Kodaxil string to reveal its layout, it looks as follows:

Fce8 3mlJ jkeN hjV7 t6ya g3p0 jkeN hM6U Sm9obg== n88r kj4r hM6 QXJtc2J5 n88r i8ux i8ux

where:

- ‘en-US’ specifies the default language and character set of the text for the whole document; here, “en-US” and “ISO-8859-1.” These are found in the default (base) thesaurus.
 - If a specific string needs a different encoding, it will be specified immediately after ‘hM6U’, a reserved word for [] ‘text’ itself.
- The parser is aware of the length of each word, element, or construct.

In the above example, “Sm9obg==” (“John”) and ‘QXJtc2J5’ (“Armsby”) -FirstName and LastName respectively- may have been extracted from a database, a web form, or another input method, and they have been base64-encoded. In Kodaxil terminology, they are .

Note that the Kodaxil internal representation is NameFirst, NameLast, as the grammar implies that the main item is prevalent. FirstName isA Name, so the internal representation is ‘NameFirst’, found in the business object base library. Local versions may specify FirstName, First_name or firstName, but ultimately they will all point to the Kodaxil word for NameFirst, so that every schema, database, markup, text or multimedia document that invokes NameFirst will use this Kodaxil word. This is the Kodaxil way to vanquish data fragmentation.

Case Study: Quantity, a Kodaxil Object Example

Elements frequently found in data interchange applications in IT are **identity**, **percentage**, and **quantity**.

A survey of existing frameworks - including uncefact, Framenet, toga, ebXml, UDEF, UDR, Cyc - shows that most of them consider time, money, and other quantities to be different entities. But from a formal standpoint there is no difference between temperature expressed in Celsius, Fahrenheit, or Kelvin, currency expressed in dollars or yen, or distance expressed in meters or yards.

Quantities

The ‘**quantity**’ Kodaxil object allows expressing numeric data in the same way, regardless of whether this data refers to money, temperature, or pairs of gloves. Also, the value is stored in a generic form that allows

converting from and to any measurement system. A breakdown of the ‘quantity’ object is out of the scope of this document.

A temperature such as 98.6 F will be expressed as: “**gMe2hTTyyGn6+PYgMf3**”

The result:

- **Uses 19 bytes;**
- Can be understood **in various languages** by means of Kodaxil tools;
- Amalgamates various semantics into the same construct (augmented information);
- Can be stored in a database where the augmented information reduces storage size;
- **Allows for automatic equivalence between systems (imperial, metric, etc.).**

The matching XML construct **uses 52 bytes**, and can be understood by English speakers only:

```
<temperature unit="degree Fahrenheit" value="98.6"/>
```

The alternate representation without attributes uses 78 bytes, and can be understood by English speakers only:

```
<temperature>  
  <unit>degree Fahrenheit</unit>  
  <value>98.6</value>  
</temperature>
```

Kodaxil internal representation uses the International System of Units (SI), with conversion modules to and from the Imperial system, and provisions for non-conventional units.

After showing that Kodaxil can represent XML constructs uniformly, one can state that Kodaxil may represent all XML constructs, build libraries of business objects, individual and aggregate, so they are equally understood worldwide by means of Kodaxil tools and thesauri.

Ontologies and Their Importance for the Semantic Web and for IT in General

The problem with terms such as Business Process management, Web Services, Semantic Web, and Ontologies is that they don’t always mean the same thing to everybody. Thus we need to make sure that definitions are alike in everybody’s mind, otherwise no language has a raison d’être.

We are happy to enlarge our definition of ontology with other people’s collaborative feedback. As stated, an ontology is simply a definition of a domain; for instance, a library can be considered a universe, with objects and methods, events, and states (‘classes’): first, we define the reason why a library exists, which is to allow people to borrow books, and to return them on a specified date. We have to define books and group them by category, subcategory, etc. (e.g. Cartoons, Anatomy, Anatomy=>Digestive system).

Categories lead to taxonomies, i.e. hierarchical organizations, as in Anatomy=>Digestive System. Note that we must take into account those cases where simple hierarchy is distorted because the same topic may belong to more than one subcategory. We must also specify rules that define the number of days an item can be borrowed, daily fees for items returned after the specified date, etc. In this simple model, ontologies are a practical means to conceptualize a domain and eventually to express it in computer format.

Ontology Matching

Ontology matching is the attempt to determine correspondences, i.e. similarities between concepts, and to find "semantic equivalences". In order to integrate heterogeneous databases that have been using different naming conventions and different names for the same business objects, there was a need to find "common semantic vectors". Using universal words as building blocks seems to solve this problem, as their building blocks refer to the same concepts.

Extensibility Implemented Through New or Aggregate Words, Thesauri, and Contexts

Each IT application is an ontology per se: it uses words to define objects, their usage, and all relations between them. It can be modeled using Kodaxil to create words, contexts, and thesauri on the fly, which are used as a dictionary for the application.

Extensibility here means that there is no limit to Kodaxil words. One can create new words, belonging to any part of speech, or aggregate words as in 'Display_Page', and create dictionaries for applications. A program can use words defined in this application dictionary, and/or use pre-defined words as business objects from a standard "universal" business library, since business processes themselves are universal and let envision generic (customizable) business libraries for use worldwide. Note that there is no limit to the volume of information each individual context can enclose. The base context, which will be open-sourced, comprises a Kodaxil-encoded dictionary, for use on a planet-wide scale.

Ontology and IT Applications

Words contain information about which part of speech they belong to, as well as their grammatical form. This helps define use cases and determine entities within an application. A dictionary then knows that 'Display_Page', using the imperative mode of an action verb, associated with a noun complement, is a process. As this dictionary knows about the lexical class of any object within its boundaries, it helps compose Model Driven Applications for use worldwide. For instance, the same MDA can be used offshore, and the same source code can be understood anywhere. Also, MDA applications can generate source code (Rational Rose, Enterprise Architect). And using Kodaxil to map statements in some programming language lets generate code that developers can understand.

The Untold Truth about the Semantic Web

Few articles and books about the semantic web mention the fact that everything that can be found on the web must be defined for it to be "usable", i.e. structured in a way that external agents have a possibility to infer on this information. This means that "somebody" should describe every object their web site mentions, from Barbie dolls to CCM skates, from Martin Luther to Bode Miller, from the Punic wars to the Coronation of Napoleon, in short a huge history and ontology of the world to date, in a modern electronic catalog.

The semantic web group places the responsibility on publishers of web sites: the information should add intelligence to contents and, by extension, media via metadata, and "at least" be expressed in RDF format, eventually using the N3 notation, and provide a deductive approach. For one, we're back to the metadata dilemma again (see " "). Then, here again, what language should the metadata use, as the group confesses that it aims " "??

Worldwide Open Semantic Tools

Wouldn't it be easier for all web developers, software architects, and users of business libraries to use the same building blocks, dragging their components from the same box? This "common pot" would enable web

developers worldwide to write web sites without any effort other than composing text in their own language, and let common tools convert it to Kodaxil words. It would also allow information seekers to search for terms and receive links in any language.

This proposal, coupled with separating handles from names on the internet -an interesting proposal by MJ O'Donnell¹²- would really make the web universal and open to other languages.

Other Usage Cases - Application to Databases

- This representation lets mix tags and semantics. This is important because column names can be understood across databases and in various languages, including flavors of the same language that cause data fragmentation. It also offers a concise content, thus saving database space.
- Since all information is base64-encoded, there is no longer a need to use data types using double-bytes such as 'nvarchar'. This encoding hides all natural-language information, such as character set, pushes it at both extremities (encoding/decoding), and eliminates it from data processing and exchange.
- The layout of the 'quantity' object makes it universal, able to represent money, distance, or pairs of gloves in the same, measurement system independent way, thus guaranteeing the highest degree of integration.
- Access on string keys is faster as keys expressed as strings are smaller (usually four sextets, occupying four bytes) for nouns in the base lexicon, five if an aggregate, preceded by a short context name as namespace.
- Datasets expressed using the same building blocks, prepared for data mining, can include data from various divisions and eventually branches in foreign countries. As a result, they are much larger, allowing for more accurate diagnostics.
- As batches of data returned from a database query expressed as Kodaxil markup are three times smaller than their XML counterparts, they save bandwidth when serialized over a network.

Application to Telephones

One can envision that a generic base thesaurus is stored in a cellular phone, and text messages are encoded as Kodaxil words. This allows composing a message in any language and having the recipient receive it in the language set on his/her phone.

Multilingual Dictionaries

Kodaxil offers the possibility to create multilingual dictionaries in all languages, including variants of words.

Indexing Binary Data

A look at the way Kodaxil sextets are encoded indicates that the binary content of a picture or sound information can be represented using Kodaxil words. One can then isolate "visual" or "sound" patterns as "sentences" and index them, making it possible to search for them using such sentences. This requires a bit of work, mathematical transforms, and alignment at word boundaries to keep important information coherent, without a shift of even one bit of information.

Web

Any web content expressed using Kodaxil words can be indexed by search engines. For information seekers, this fact allows the same terms to gather information from multiple, multiplies the results, as searches operate on data previously encoded (from natural language to Kodaxil). This helps finding more -or more accurate- information about the same search topic. A search engine turning words from various languages into Kodaxil words is under development.

A Worldwide Dictionary

A Kodaxil-encoded dictionary lets express the meaning of a word, whether general (common to all humans) or specific to a natural language. This includes derivative forms, which are currently handled by very few dictionaries.

A Worldwide Wiki

A Kodaxil-encoded wiki lets anybody around the world store information in a common space, contributing to the most formidable international effort of building a huge knowledge base for mankind.

CONCLUSION

Kodaxil addresses global semantic interoperability and construction of common business objects that can be reused on a planet-wide scale, thus affecting all domains of IT: it can be applied to text analysis, web and database contents and storage, data mining, entity extraction, ETL, integration of IT operations in mergers and acquisitions, semantic-aware web services, multilingual or conceptual search engines, multilingual ontologies, knowledge management and collective intelligence sharing, programming languages and operating systems, and model-driven architectures. It brings a semantic interoperability that lets one envision the design of web services with contexts, described in various languages that can be integrated. People speaking different languages may use it to converse in real time, e.g. in a control tower, or to spread emergency information to speakers of multiple languages. With regard to applications, one can use it to build knowledge-based systems, in expert arenas such as risk analysis and assessment, rendering it particularly useful in an outsourcing context.

In medicine, Kodaxil allows to systematize the representation of medical knowledge in order to provide accurate primary and differential diagnoses, for the purpose of teaching and learning clinical medicine by placing students 'in situ' -in the same way pilots practice with flight simulators- and for disease control and prevention (practical day-to-day use). Medical search engines can benefit from Kodaxil when the representation of medical knowledge has been defined. Another project has been initiated to allow fast retrieval of complex information (search engines by concept).

Much work remains to be done on this technology, especially with regard to building business objects and processes libraries for use worldwide, as business objects require less 'literary' and semantic information in contexts than text, or full-fledge reasoning systems. Whether this work will occur or not depends on the system's merits, and to the set of tools that help migrate from XML to this new representation.

The open source movement will also help the set of base words to evolve towards a full-fledge universal instrument for project and idea interchanges.

End Notes

1. 5th International Conference on Business Process Management [<http://bpm07.fit.qut.edu.au>].
2. Enterprise Integration: The Essential Guide to Integration Solutions. Beth Gold-Bernstein, William Ruh. Addison-Wesley: Boston. 2005.
- Journal, 17, 93-107.
3. Data Mining: Introductory and Advanced Topics. Margaret Dunham. Prentice Hall: Upper Saddle River NJ. 2003.
4. Liebeskind, J.P. (1996). 'Knowledge, strategy, and the theory of the firm', Strategic Management
5. The Big Book of Concepts. Gregory L. Murphy. MIT Press: Cambridge MA. 2002.
6. The World is a Text: The Writing, Reading, and Thinking About Culture and Its Contexts. 2nd edition. Jonathan Silverman, Dean Rader. Prentice Hall: Upper Saddle River NJ. 2006.
7. Linguistic Simulation of Semantic Invariants for Multilingual Knowledge Management Systems. Elena Kozerenko.
8. Duraljan Vocabulary: Lexical Similarities in the Major Agglutinative Languages. Panu Hannu Aukusti Hakola. HPA Hakola: Kuopio. 1997.
- Institute for Problems of Informatics of the Russian Academy of Sciences: Moscow.
9. The Architecture of Language. Noam Chomsky. Oxford University Press: New York. 2006.
10. Taking the Census of English Words. Robert L. Ramsay. American Speech, Vol. 8, No. 1 (Feb. 1933), pp. 36-41.
11. DARPA. Linguistic Data Consortium program
12. Separate handles from names on the internet By MJ O'Donnel - Communications of the ACM Dec 2005 Vol 48 no 12